HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P. O. Box 272400
Fort Collins, Colorado 80527-2400

PATENT APPLICATION
ATTORNEY DOCKET NO. 10007965-1

IN THE
UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s): Jun Li et al.

Confirmation No.: 9833

Application No.: 09/955,764

Examiner: Rutten, James D.

Filing Date: Sep. 19, 2001

Group Art Unit: 2192

Title: RUN-TIME MONITORING IN COMPONENT-BASED SYSTEMS

Mail Stop Appeal Brief-Patents
Commissioner For Patents
PO Box 1450
Alexandria, VA 22313-1450

TRANSMITTAL OF APPEAL BRIEF

Sir:

Transmitted herewith is the Appeal Brief in this application with respect to the Notice of Appeal filed on Sep. 6, 2006 and in response to the Notice of Panel Decision from Pre-Appeal Brief Review dated November 24, 2006.

The fee for filing this Appeal Brief is (37 CFR 1.17(c)) $500.00.

(complete (a) or (b) as applicable)

The proceedings herein are for a patent application and the provisions of 37 CFR 1.136(a) apply.

( ) (a) Applicant petitions for an extension of time under 37 CFR 1.136 (fees: 37 CFR 1.17(a)-(d) for the total number of months checked below:

( )	one month	$120.00
( )	two months	$450.00
( )	three months	$1020.00
( )	four months	$1590.00

( ) The extension fee has already been filed in this application.

(X) (b) Applicant believes that no extension of time is required. However, this conditional petition is being made to provide for the possibility that applicant has inadvertently overlooked the need for a petition and fee for extension of time.

Please charge to Deposit Account 08-2025 the sum of $500.00. At any time during the pendency of this application, please charge any fees required or credit any over payment to Deposit Account 08-2025 pursuant to 37 CFR 1.25. Additionally please charge any fees to Deposit Account 08-2025 under 37 CFR 1.16 through 1.21 inclusive, and any other sections in Title 37 of the Code of Federal Regulations that may regulate fees. A duplicate copy of this sheet is enclosed.

( ) I hereby certify that this correspondence is being deposited with the United States Postal Service as first class mail in an envelope addressed to: Commissioner for Patents, Alexandria, VA 22313-1450. Date of Deposit:_____
OR

(X) I hereby certify that this paper is being transmitted to the Patent and Trademark Office facsimile number (571) 273-8300 on Dec. 26, 2006

Number of pages: 31

Typed Name: Michaela Pratt

Signature: /Michaela Pratt/

Respectfully submitted,

Jun Li et al.

By /T A K/

Timothy B. Kang

Attorney/Agent for Applicant(s)
Reg. No. 46,423

Date: Dec. 26, 2006

PATENT
Atty Docket No.: 10007965-1

In The U.S. Patent and Trademark Office

In Re the Application of:

Inventor(s):	Jun Li et al.	**Confirmation No.:**	9833
Serial No.:	09/955,764	**Examiner:**	Rutten, James D.
Filed:	September 19, 2001	**Group Art Unit:**	2192
Title:	RUN-TIME MONITORING IN COMPONENT-BASED SYSTEMS		

MAIL STOP AMENDMENT
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

CERTIFICATE OF FACSIMILE TO THE USPTO

I hereby certify that this correspondence is being transmitted to the Patent and Trademark Office facsimile number (571) 273-8300 on December 26, 2006. This correspondence contains the following document(s):

1 sheet of Transmittal of Appeal Brief (2 copies).

28 sheets of Appeal Brief - Patents

Respectfully submitted,

MANNAVA & KANG, P.C.

December 26, 2006

Timothy B. Kang
Registration No.: 46,423

MANNAVA & KANG, P.C.
8221 Old Courthouse Road
Suite 104
Vienna, VA 22182
(703) 652-3817
(703) 865-5150 (facsimile)

RECEIVED
CENTRAL FAX CENTER

DEC 2 6 2006

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
P.O. Box 272400
Fort Collins, Colorado 80527-2400

Attorney Docket No.: 10007965-1

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Inventor(s):	JUN LI et al.	**Confirmation No.:**	9833
Serial No.:	09/955,764	**Examiner:**	James D. Rutten
Filed:	September 19, 2001	**Group Art Unit:**	2192
Title:	RUN-TIME MONITORING IN COMPONENT-BASED SYSTEMS		

MAIL STOP APPEAL BRIEF - PATENTS
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF - PATENTS

Sir:

This is an Appeal Brief in connection with the decisions of the Notice of Panel Decision from Pre-Appeal Brief Review dated November 24, 2006 and of the Examiner in a final Office Action dated June 6, 2006. It is respectfully submitted that the present application has been more than twice rejected. Each of the topics required in an Appeal Brief and a Table of Contents are presented herewith and labeled appropriately.

This Appeal Brief is hereby submitted within one month of the Notice of Panel Decision from Pre-Appeal Brief Review because December 26th is the first business day following a weekend day and a federal holiday.

12/29/2006 EFLORES 00000014 082025 09955764
01 FC:1402 500.00 DA

PATENT

Atty Docket No.: 10007965-1
App. Ser. No.: 09/955,764

TABLE OF CONTENTS

PATENT

Atty Docket No.: 10007965-1
App. Ser. No.: 09/955,764

(1) Real Party In Interest

The real party in interest is Hewlett-Packard Development Company, L.P.

(2) Related Appeals And Interferences

There are no other appeals or interferences related to this case.

(3) Status Of Claims

Claims 1-43 are pending and rejected. All pending Claims 1-43 are hereby appealed.

(4) Status of Amendments

No amendment was filed subsequent to the final Office Action dated June 6, 2006.

(5) Summary Of Claimed Subject Matter

According to one embodiment in Claim 1, there is provided a monitoring method for a component-based software system operating over one or more processing devices, comprising the steps of:

initiating an invocation of a second software component from within an execution of a first software component (Parag. [0047]; 207, FIG. 2);

recording a stub start log data including a global causal identifier in an instrumented stub before said invocation of the second software component (Parags. [0043] and [0057]);

transmitting the global causal identifier from the first software component to the second software component, where the second software component executes on a separate thread and in a system remote from the first software component (Parag. [0057]);

recording a stub end log data including the global causal identifier in the instrumented stub after a response is received from the invocation of the second software component, the response including the global causal identifier (Parag. [0046]; 245, FIG. 2);

wherein the stub start log data and said stub end log data gather runtime information about execution of the second software component within the component-based software system (Parags. [0033] and [0037]).

Furthermore, in Claim 7 (dependent on Claim 1) a log data content is configured during operation of the component-based software system (Parag. [0055]).

Furthermore, in Claim 8 (dependent on Claim 7) a runtime information generated during the operation of the component-based software system includes a regular expression that determines a particular log data contents, and wherein a user is capable of changing the particular log data contents during the operation of the component-based software system by setting the regular expression (Parag. [0055].

According to another embodiment in Claim 29, there is provided a monitoring method for a component-based software system operating over one or more processing devices, comprising the steps of:

accumulating one or more stub start log data entries including a global causal identifier wherein the global causal identifier is transmitted from a first software component to a second

software component executing on a separate thread and in a system remote from the first software component, with a stub start log data entry of said one or more stub start data entries being recorded by an instrumented stub before a subsequent software component invocation (Parag. [0057];

accumulating one or more skeleton start log data entries including the global causal identifier, with a skeleton start log data entry of said one or more skeleton start data entries being recorded by an instrumented skeleton before the instrumented skeleton invokes the subsequent software component (Parag. [0044]; 236, FIG. 2);

accumulating one or more skeleton end log data entries including the global causal identifier, with a skeleton end log data entry of the one or more skeleton end log data entries being recorded by the instrumented skeleton after a response is received from the subsequent software component invocation (Parags. [0045] and [0046]; 240 and 245, FIG. 2);

accumulating one or more stub end log data entries including the global causal identifier, with a stub end log data entry of said one or more stub end log data entries being recorded by the instrumented stub after the response is received from the subsequent software component invocation (Parags. [0046] and [0057]); and

processing an accumulated log data, using the global causal identifier, and calculating a system behavior characteristic for one or more software components executing within the component-based software system (Parag. [0061]).

According to still another embodiment in **Claim 36**, there is provided a computer system adapted to monitor component-based software applications, comprising:

at least one processing device residing in said computer system; one or more software components residing on the at least one processing device and capable of executing in the computer system (Parag. [0038]);

at least one other processing device residing remotely from the computer system, the at least one other processing device having one or more software components residing therein (Parag. [0038]); and

one or more instrumented stubs in said one or more software components, with an instrumented stub being capable of recording a stub start log data at an execution invocation of the instrumented stub in a first software component and recording a stub end log data at an execution conclusion of the instrumented stub, the stub start log data and the stub end log data including a global causal identifier and wherein the one or more instrumented stubs is configured to transmit the global causal identifier from one of the software components in the at least one processing device to at least one other component in the at least one other processing device (Parag. [0043]; Probes, FIG. 3).

(6) Grounds of Rejection to be Reviewed on Appeal

a) Whether Claims 1-7, 9-11, 13-19, 21-32, and 35-42 should have been rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Kazi et al. ("JaViz: A Client/Server Java Profiling Tool") in view of Blumson et al. ("Automatic Insertion of Performance Instrumentation for Distributed Applications") and further in view of Delucia et al. (U.S. Patent Number 4,819,233) and Tucker et al. (U.S. Patent Number 6,151,639).

b) Whether Claims 8, 12, and 43 should have been rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Kazi et al., Blumson et al., Delucia et al., and Tucker et al. as applied to claims 7, 9, and 36, respectively, and further in view of Courant et al. (U.S. Patent Number 5,522,073).

c) Whether Claim 20 should have been rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Kazi et al., Blumson et al., Delucia et al., and Tucker et al. as applied to claim 9 and further in view of Brandle et al. (U.S. Patent Number 5,146,593).

d) Whether Claims 33 and 34 should have been rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over Kazi et al., Blumson et al., Delucia et al., and Tucker et al. as applied to claim 29 and further in view of Peek et al. ("Unix Power Tools").

(7) **Arguments**

A. The rejection of Claims 1-7, 9-11, 13-19, 21-32, and 35-42 under 35 U.S.C. § 103(a) as allegedly being unpatentable over Kazi et al. in view of Blumson et al. and further in view of Delucia et al. and Tucker et al. is improper

The test for determining if a claim is rendered obvious by one or more references for purposes of a rejection under 35 U.S.C. § 103 is set forth in MPEP § 706.02(j):

> To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. The

> teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art and not based on applicant's disclosure. *In re Vaeck*, 947 F.2d 488, 20 USPQ2d 1438 (Fed. Cir. 1991).

Therefore, if the above-identified criteria are not met, then the cited reference(s) fails to render obvious the claimed invention and, thus, the claimed invention is distinguishable over the cited reference(s).

Independent Claims 1, 29, and 36

Independent Claims 1, 29, and 36 recite, "recording a stub start log data ... in an instrumented stub" and "transmitting the global causal identifier from the first software component to the second software component." The Examiner has alleged that Kazi shows: a) a method identifier as the claimed global causal identifier; b) a first method in Jvm 1 (Kazi's Figure 3) as the claimed first software component; c) a second method in Jvm 2 (Kazi's Figure 3) as the claimed second software component that can be called and executed by the first method in Jvm 1; and d) a record in a .jta file as the claimed instrumented stub (the Examiner cited to Kazi's p. 7, "Detailed trace generation" that states "The trace generation module of the Jvm is modified to record every invocation of a method;" whereby Kazi's p.5, second full paragraph, under "Tree generation" states that such a record is written to an output .jta file). *See* Final Rejection, starting at p. 5.

First, the .jta file cannot be the "instrumented stub" as claimed and so defined in at least paragraph [0027] of the present application because the .jta file is merely a data file. Second, as

pointed out by the Examiner, Kazi's p. 9, third paragraph, discloses that each unique method identifier (alleged as the claimed "global causal identifier) associated with the method to be invoked is *given by a RMI module at the server Jvm*, that is, Jvm 2. Thus, this method identifier belongs to the method associated with the "second software component" (in Jvm 2) and is recognizable by the "first software component" (in Jvm 1), after Jvm 1 discovers the second software component (via some object discovery protocol), and before the method invocation from Jvm 1 to Jvm 2 happens. That is, the method identifier creation is at the second software component (in Jvm 2), not at the first software component (in Jvm 1).

In contrast, the claimed global causal identifier is designed for the end-to-end call tracking purpose (for instance, a call chain that can have more than one call involved, spanning through more than one machine), independent of the underlying remote object invocation protocol. Each call chain shares the same global causal identifier, which is created by the chain initiator (the first component). That is, in Claim 1, the global causal identifier is CREATED by component 1, and then TRANSMITTED to component 2. Component 1, rather than component 2, is the SOURCE of the creation of the global causal identifier.

As cited by the Examiner, although Tucker '639 provides the transmission of a system-wide identifier in a remote object invocation request to an appropriate remote node (Tucker's col. 3, ll. 22-24), Tucker cannot be combined with Kazi to show the transmission of a method identifier *from* Jvm 1 *to* Jvm 2 when, as Kazi states, the method identifier actually originates from Jvm 2. Tucker's system-wide identifier is designed for the purpose of remote object invocation, specifically, to locate which particular method in which objects in a remote machine

should be invoked. Such system-wide identifiers also exist in Kazi's system (called RMI), in order to carry out remote object invocation. However, such a remote-object-invocation-oriented unique identifier is different from the claimed global causal identifier that is designed for "call causality tracking." These system-wide identifiers are not designed to take into account the logging of the ordering of the calls incurred to the same target software component from multiple other (client) components. Accordingly, Tucker '639 does not cure the defect in Kazi, and the proposed combination of Tucker and Kazi would neither disclose nor make obvious all of the elements claimed in claims 1, 29, and 36.

In addition, Blumson and Delucia '233 also fail to make up for the above-described deficiencies in Kazi and Tucker '639.

Accordingly, it is respectfully submitted that the Examiner failed to establish a *prima facie* case of obviousness against independent Claims 1, 29, and 36. Withdrawal of the rejection of these claims and their allowance are therefore requested. Consequently, the dependent claims 2-28, 30-35, and 37-43 are also allowable by virtue of the allowability of their independent Claims 1, 29, and 36.

Claim 7

Claim 7 recites, "log data contents is configured during operation of said component-based software system." The Examiner cited to a visualizer in Kazi to allege coverage of such claimed recitation. As cited by the Examiner in the Office Action dated June 7, 2004, Kazi's p. 5, paragraph 5, discusses the use of a visualizer that reads the .jta output file *of the post-processing step* for a graphic display of a generated tree of trace files. Therefore, any log data

contents being configured in the visualizer is configured *after* the operation of the software system in Kazi, and not *during* such an operation as claimed.

Accordingly, it is respectfully submitted that the Examiner has failed to establish a *prima facie* case of obviousness against claim 7 for reasons in addition to the reasons set forth earlier regarding independent claim 1. Consequently, claim 7 is allowable over the references of record for at least the above reasons.

B. The rejection of Claims 8, 12, and 43 under 35 U.S.C. § 103(a) as allegedly being unpatentable over Kazi et al., Blumson et al., Delucia et al., Tucker et al., and Courant et al. is improper

The Examiner does not and cannot reasonably assert that Courant et al. makes up for the deficiencies in Kazi et al., Blumson et al., Delucia et al., and Tucker et al. discussed above. It is thus respectfully submitted that the Examiner has failed to establish a *prima facie* case of obviousness against claims 8, 12, and 43 for reasons in addition to the reasons set forth earlier regarding the independent claims 1 and 36. Consequently, claims 8, 12, and 43 are allowable over the references of record for at least those reasons set forth earlier with regard to the allowability of their independent claims 1 and 36.

Claim 8

Dependent upon Claim 7, Claim 8 further recites that "a user is capable of changing said particular log data contents during said operation of said component-based software system by

setting said regular expression." In contrast, as cited by the Examiner in the Office Action dated June 7, 2004, Courant's col. 8, ll. 26-32, discusses the creation of a software routine for execution or operation in a software system, wherein a user is allowed to modify various parameters for the routine prior to its execution or operation. In other words, Courant is concerned with the *pre-processing step* of creating a software system for subsequent operation thereof, whereby user modification is possible *prior* to the operation of such a software system, and not *during* such an operation as claimed. In fact, Courant's regular expression is defined for the purpose of triggering the execution of other software routine in another software tool, to respond to the event raised by a different software tool. Dynamic changing of regular expression to alter the responsive behavior of the software tool set is not meaningful. Furthermore, the proposed combination of Kazi and Courant would render such a combined system inoperable because Kazi's visualizer is concerned with procedures in a post-processing step and Courant is concerned with procedures in a pre-processing step that would be incompatible with Kazi's post-processing step.

Accordingly, in addition to the reasons set forth earlier regarding independent claim 1, claim 8 is allowable over the references of record for at least the above reasons.

C. The rejection of Claim 20 under 35 U.S.C. § 103(a) as allegedly being unpatentable over Kazi et al., Blumson et al., Delucia et al., Tucker et al., and Brandle et al. is improper

The Examiner does not and cannot reasonably assert that Brandle et al. makes up for the deficiencies in Kazi et al., Blumson et al., Delucia et al., and Tucker et al. discussed above. It is

thus respectfully submitted that the Examiner has failed to establish a *prima facie* case of obviousness against claim 20 for reasons in addition to the reasons set forth earlier regarding the independent claim 1. Consequently, claim 20 is allowable over the references of record for at least those reasons set forth earlier with regard to the allowability of its independent claim 1.

D. The rejection of Claims 33 and 34 under 35 U.S.C. § 103(a) as allegedly being unpatentable over Kazi et al., Blumson et al., Delucia et al., Tucker et al., and Peek et al. is improper

The Examiner does not and cannot reasonably assert that Peek et al. makes up for the deficiencies in Kazi et al., Blumson et al., Delucia et al., and Tucker et al. discussed above. It is thus respectfully submitted that the Examiner has failed to establish a *prima facie* case of obviousness against claims 33 and 34 for reasons in addition to the reasons set forth earlier regarding the independent claim 29. Consequently, claims 33 and 34 are allowable over the references of record for at least those reasons set forth earlier with regard to the allowability of their independent claim 29.

(8) Conclusion

For at least the reasons given above, the rejection of claims 1-43 is improper. Accordingly, it is respectfully requested that such a rejection by the examiner be reversed and these claims be allowed. Attached below for the Board's convenience is an Appendix of claims 1-43 as currently pending and on appeal.

PATENT

Atty Docket No.: 10007965-1
App. Ser. No.: 09/955,764

Please grant any required extensions of time and charge any fees due in connection with this Appeal Brief to deposit account no. 08-2025.

Respectfully submitted,

Dated: December 26, 2006 By

Tiep H. Nguyen
Registration No. 44,465

Timothy B. Kang
Registration No.: 46,423

MANNAVA & KANG, P.C.
8221 Old Courthouse Road
Suite 104
Vienna, VA 22182
(703) 652-3822 (or 3817)
(703) 880-5270 (facsimile)

(9) Claim Appendix

1. A monitoring method for a component-based software system operating over one or more processing devices, comprising the steps of:

initiating an invocation of a second software component from within an execution of a first software component;

recording a stub start log data including a global causal identifier in an instrumented stub before said invocation of said second software component;

transmitting the global causal identifier from the first software component to the second software component wherein the second software component executes on a separate thread and in a system remote from the first software component;

recording a stub end log data including the global causal identifier in said instrumented stub after a response is received from said invocation of said second software component, said response including the global causal identifier;

wherein said stub start log data and said stub end log data gather runtime information about execution of said second software component within said component-based software system.

2. The method of claim 1, wherein said instrumented stub is generated from a description of an interface of said second software component.

3. The method of claim 1, wherein said second software component is remote from said first software component.

4. The method of claim 1, wherein said first software component resides on a first processing device and said second software component resides on a second processing device.

5. The method of claim 1, further comprising the preliminary step of selecting a log data contents to be included in said stub start and stub end log data, with the selecting step logging zero or more of an application semantic behavior data, a timing latency data, a shared resource usage data, and a causality relationship data.

6. The method of claim 1, wherein a log data contents is configured during generation of said instrumented stub.

7. The method of claim 1, wherein a log data content is configured during operation of said component-based software system.

8. The method of claim 7, wherein a runtime information generated during said operation of said component-based software system includes a regular expression that determines a particular log data contents, and wherein a user is capable of changing said particular log data contents during said operation of said component-based software system by setting said regular expression.

9. The method of claim 1, further comprising the steps of:

initiating said invocation of said second software component from within an execution of an instrumented skeleton;

recording a skeleton start log data before said instrumented skeleton invokes said second software component; and

recording a skeleton end log data in said instrumented skeleton after a response is received from said invocation of said second software component.

10. The method of claim 9, wherein said instrumented skeleton is generated from a description of an interface of said second software component.

11. The method of claim 9, wherein said instrumented skeleton is generated from a description of an interface of said second software component and wherein said second software component is remote from said first software component.

12. The method of claim 9, wherein a particular instrumented stub is capable of enabling and disabling a data logging capability of a corresponding instrumented skeleton.

13. The method of claim 9, wherein an accumulated log data from a plurality of instrumented stubs and a plurality of instrumented skeletons is collected and correlated.

14. The method of claim 9, wherein said stub start, stub end, skeleton start, and skeleton end log data capture a causality relationship data between said first software component and said second software component.

15. The method of claim 9, wherein said stub start, stub end, skeleton start, and skeleton end log data are used to determine a causality relationship data for a plurality of threads.

16. The method of claim 9, wherein said stub start, stub end, skeleton start, and skeleton end log data are used to determine a causality relationship data for a plurality of threads spawned during invocation of said second software component.

17. The method of claim 9, wherein said stub start, stub end, skeleton start, and skeleton end log data are used to determine a causality relationship data for a thread in which said first software component is invoked.

18. The method of claim 9, further comprising the preliminary step of selecting a log data contents to be included in said skeleton start and skeleton end log data, with the selecting step logging zero or more of a timing latency data, a shared resource usage data, and a causality relationship data.

19. The method of claim 9, wherein the method includes a transportation of at least a portion of said stub start log data of said instrumented stub to said instrumented skeleton.

20. The method of claim 19, wherein said transportation is accomplished by passing the global causal identifier to a function defined in an interface definition of said second software component.

21. The method of claim 9, wherein said instrumented skeleton stores at least a portion of said skeleton start log data to a thread-specific storage.

22. The method of claim 21, wherein an event number included in said at least a portion of said skeleton start log data is updated before being copied into said thread-specific storage.

23. The method of claim 9, further comprising the steps of:

retrieving a thread-transportable log data from a thread-specific storage of a parent thread;

transporting said thread-transportable log data to a child thread;

adding a thread information about a child thread to said thread-transportable log data to form a child thread data; and

recording said child thread data to a thread table of said child thread.

24. The method of claim 23, wherein said thread-transportable log data comprises a self thread identifier and optionally a function container identifier, with said self thread identifier distinguishing user-application generated threads from threads generated by an underlying component-based system runtime infrastructure.

25. The method of claim 9, further comprising the step of intercepting dynamic memory allocation and de-allocation requests and logging a heap memory usage data from said requests.

26. The method of claim 9, wherein a particular log data is recorded in a per- process log table.

27. The method of claim 9, wherein a particular log data is recorded on a per-thread basis.

28. The method of claim 9, wherein a particular log data is stored in a persistent storage.

29. A monitoring method for a component-based software system operating over one or more processing devices, comprising the steps of:

accumulating one or more stub start log data entries including a global causal identifier wherein the global causal identifier is transmitted from a first software component to a second software component executing on a separate thread and in a system remote from the first software component, with a stub start log data entry of said one or more stub start data entries being recorded by an instrumented stub before a subsequent software component invocation;

accumulating one or more skeleton start log data entries including the global causal identifier, with a skeleton start log data entry of said one or more skeleton start data entries being recorded by an instrumented skeleton before said instrumented skeleton invokes said subsequent software component;

accumulating one or more skeleton end log data entries including the global causal identifier, with a skeleton end log data entry of said one or more skeleton end log data entries being recorded by said instrumented skeleton after a response is received from said subsequent software component invocation;

accumulating one or more stub end log data entries including the global causal identifier, with a stub end log data entry of said one or more stub end log data entries being recorded by said instrumented stub after said response is received from said subsequent software component invocation; and

processing an accumulated log data, using the global causal identifier, and calculating a system behavior characteristic for one or more software components executing within said component-based software system.

30. The method of claim 29, wherein said system behavior characteristic comprises a causality relationship data.

31. The method of claim 29, wherein said system behavior characteristic comprises an application semantic behavior data.

32. The method of claim 29, wherein said system behavior characteristic comprises a shared resource usage data.

33. The method of claim 29, wherein said system behavior characteristic comprises a shared resource usage data, with said shared resource usage data including a CPU usage data.

34. The method of claim 29, wherein said system behavior characteristic comprises a shared resource usage data, with said shared resource usage data including a memory usage data.

35. The method of claim 29, wherein said system behavior characteristic comprises a timing latency data.

36. A computer system adapted to monitor component-based software applications, comprising:

at least one processing device residing in said computer system; one or more software components residing on said at least one processing device and capable of executing in said computer system;

at least one other processing device residing remotely from said computer system, the at least one other processing device having one or more software components residing therein; and

one or more instrumented stubs in said one or more software components, with an instrumented stub being capable of recording a stub start log data at an execution invocation of said instrumented stub in a first software component and recording a stub end log data at an execution conclusion of said instrumented stub, said stub start log data and said stub end log data including a global causal identifier and wherein said one or more instrumented stubs is configured to transmit said global causal identifier from one of the software components in the at least one processing device to at least one other component in the at least one other processing device.

37. The system of claim 36, further comprising a memory capable of storing said stub start and stub end log data.

38. The system of claim 36, further comprising one or more instrumented skeletons, with an instrumented skeleton being capable of recording a skeleton start log data at an execution invocation of said instrumented skeleton in a second software component and recording a skeleton end log data at an execution conclusion of said instrumented skeleton.

39. The system of claim 36, wherein a first software component of said one or more software components is capable of invoking a second software component.

40. The system of claim 36, wherein a first software component of said one or more software components is capable of invoking a second software component and wherein said first software component resides on a first processing device and said second software component resides on a second processing device.

41. The system of claim 36, wherein said memory further includes a thread table adapted to store thread log data.

42. The system of claim 36, wherein said component-based software system further comprises a persistent storage capable of collecting a plurality of log data.

43. The system of claim 36, further comprising:

a persistent storage capable of collecting a plurality of log data;

an analyzer communicating with said persistent storage and capable of retrieving and analyzing log data from said persistent storage; and

a monitoring coordinator communicating with one or more instrumented, component-based software applications and capable of enabling or disabling instrumented stubs and instrumented skeletons.

PATENT

Atty Docket No.: 10007965-1
App. Ser. No.: 09/955,764

(10) Evidence Appendix

None.

PATENT

Atty Docket No.: 10007965-1
App. Ser. No.: 09/955,764

(11) Related Proceedings Appendix

None.